PHASE 1: BUSINESS UNDERSTANDING

1.1 Introduction

Solar energy is one of the most promising solutions for addressing the global energy crisis and reducing carbon emissions. However, many solar farms still struggle with inefficiencies due to fluctuating irradiance levels, hardware issues, and lack of real-time performance analysis. This project aims to leverage artificial intelligence (AI) techniques to analyze and improve solar power production using predictive and classification models.

1.2 Real-World Problem

Despite increasing investments in solar infrastructure, many energy providers lack efficient tools to monitor and optimize power output from their solar clusters. These inefficiencies can result in energy losses and reduced return on investment. Therefore, there is a critical need to analyze performance data and develop intelligent systems that can predict power generation and detect underperforming clusters.

1.3 Link to SDGs

This project supports the following United Nations Sustainable Development Goals (SDGs):

I. **Goal 7: Affordable and Clean Energy** – by enhancing solar energy efficiency and promoting renewable energy solutions.

II. **Goal 13: Climate Action** – by reducing reliance on fossil fuels and contributing to carbon footprint reduction through clean energy adoption.

1.4 Project Objectives

I. To explore and analyze solar energy data from multiple clusters.

II. To build a Random Forest regression model to predict solar power output based on irradiance and cluster behavior.

III. To develop an XGBoost classification model that classifies clusters into “High” or “Low” performance categories using efficiency ratios.

IV. To deploy all findings in a user-friendly dashboard for monitoring and decision support.

1.5 Success Criteria

I. A Streamlit-based dashboard capable of loading real Excel datasets and visualizing key metrics.

II. Regression model achieving $R^2 > 0.90$ and MSE within acceptable thresholds.

III. Classification model reaching an AUC score > 0.90 with accurate confusion matrix performance.

IV. Insightful visualizations that help stakeholders identify trends and improve operations.

1.6 Key Stakeholders

I. **Solar Energy Operators** – to detect inefficiencies and monitor cluster performance.

II. **Data Analysts and Engineers** – to interpret model predictions and optimize energy strategies.

III. **Environmental Agencies** – to promote clean energy initiatives and measure climate impact.

PHASE 2: DATA UNDERSTANDING

2.1 Dataset Description

The dataset used in this project originates from a weather station monitoring solar power production for the month of June 2024. The data is organized in a directory named "2407_Weather Station (Jun24)", which contains 30 subfolders one for each day of the month (e.g., 01062024, 02062024, ...). Each daily folder includes multiple Excel files and a reference image that capture both weather conditions and solar energy production. The key files within each folder are described below:

File Name	**Description**
Inverter Data.xls	Contains 13 sheets, each representing a solar inverter (CLUSTER 1–13). Each sheet logs Active Power (kW) every 5 minutes, serving as the main source of production data.
Weather Condition.xls	Records solar irradiance metrics, such as Average Irradiance, throughout the day. This file helps correlate production with sunlight availability.
PR.xlsx	Contains the Performance Ratio (PR) calculated hourly or periodically, reflecting system efficiency relative to irradiance.
Substation.xls	Logs output power at the substation level (likely from the AC side), representing the final delivered energy.
Total Energy.xls	Summarizes the total energy produced per day, which is essential for computing daily yield in kilowatt-hours (kWh).
Daily Energy.xls	Details the daily energy output, potentially broken down by inverter or cluster for more granular insight.
Max Load.PNG	A visual reference showing peak load (e.g., maximum current or power), useful for system diagnostics.

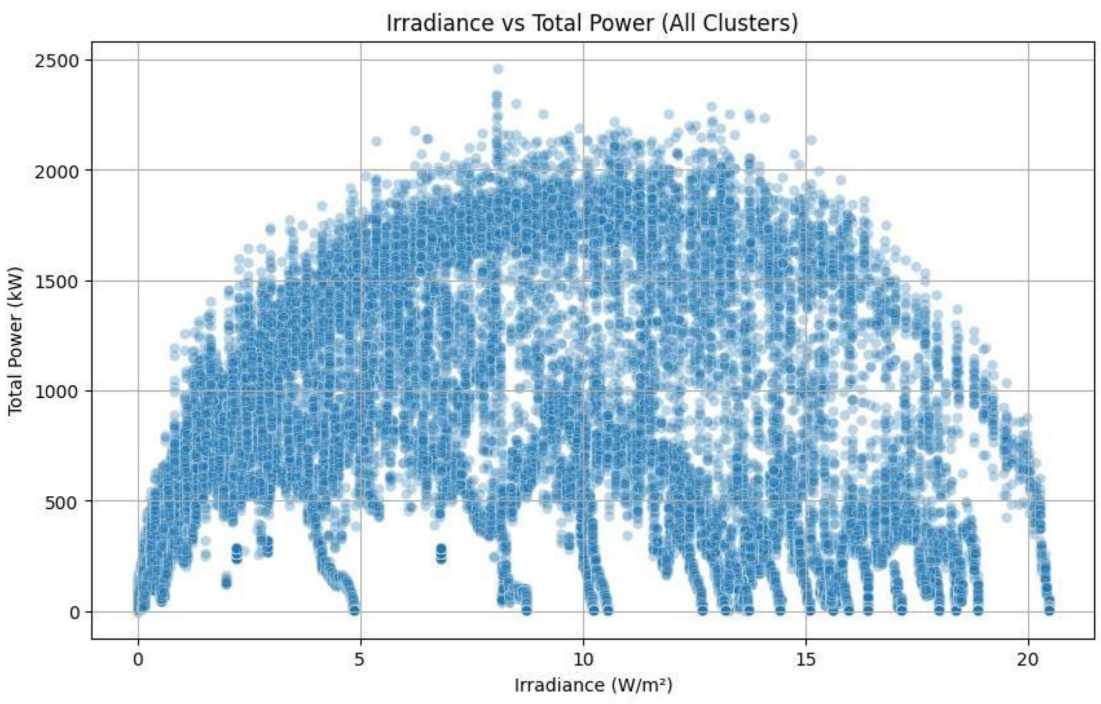
These files, collectively, provide a comprehensive view of solar power generation, system efficiency, and weather influence. For this project, the main modeling and visualization focus is on Inverter Data and Weather Condition, which are integrated to derive useful features such as efficiency and performance classification.

2.2 Initial Exploration

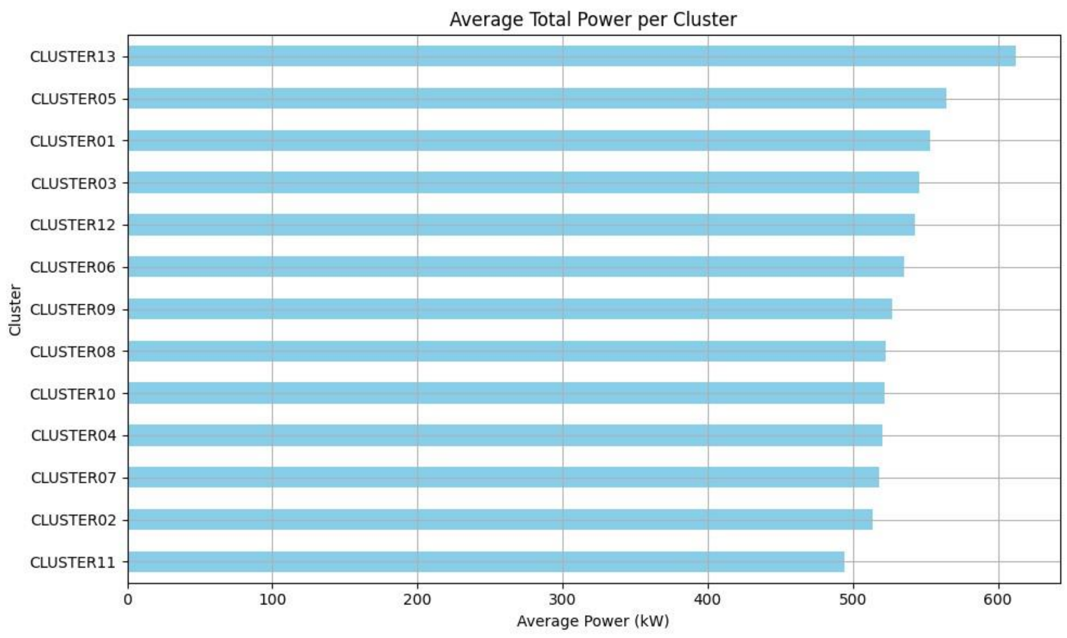
During the initial data exploration, several visual analyses were performed to uncover key patterns and relationships within the solar performance data. These visuals include scatter plots, bar charts, histograms, line charts, and box plots — all aimed at understanding the behavior of solar power output in relation to external factors such as irradiance and time.

This exploration provided insights into how clusters perform differently, how power generation varies by hour and by day, and how efficiency fluctuates across conditions. Additionally, some plots highlighted the distribution and variability of critical features like irradiance and power output.

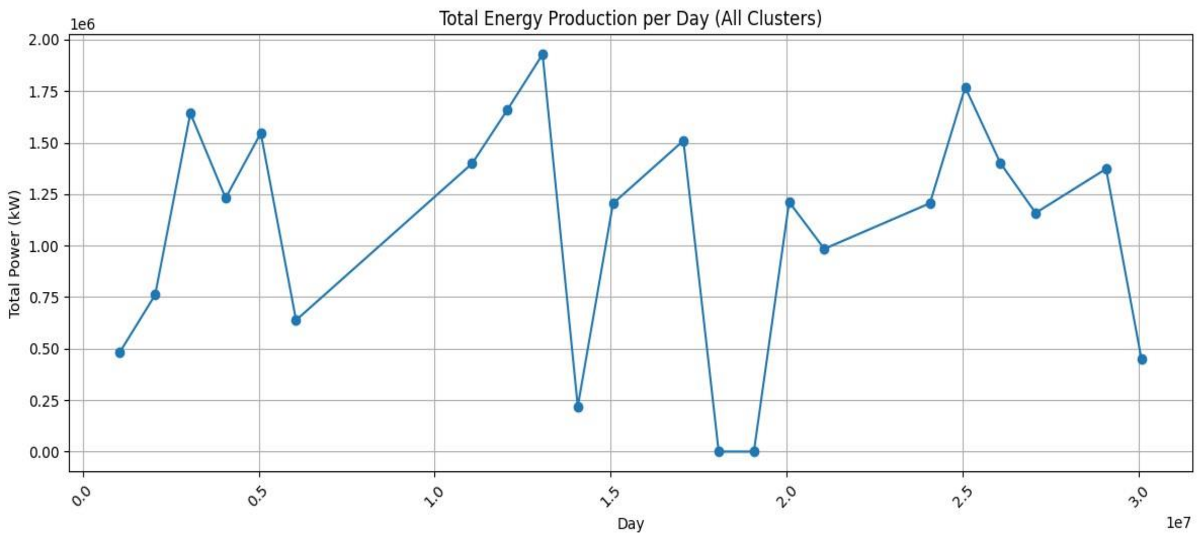
The following figures present these findings in more detail, offering a clearer picture of system performance and helping guide the development of prediction and classification models.



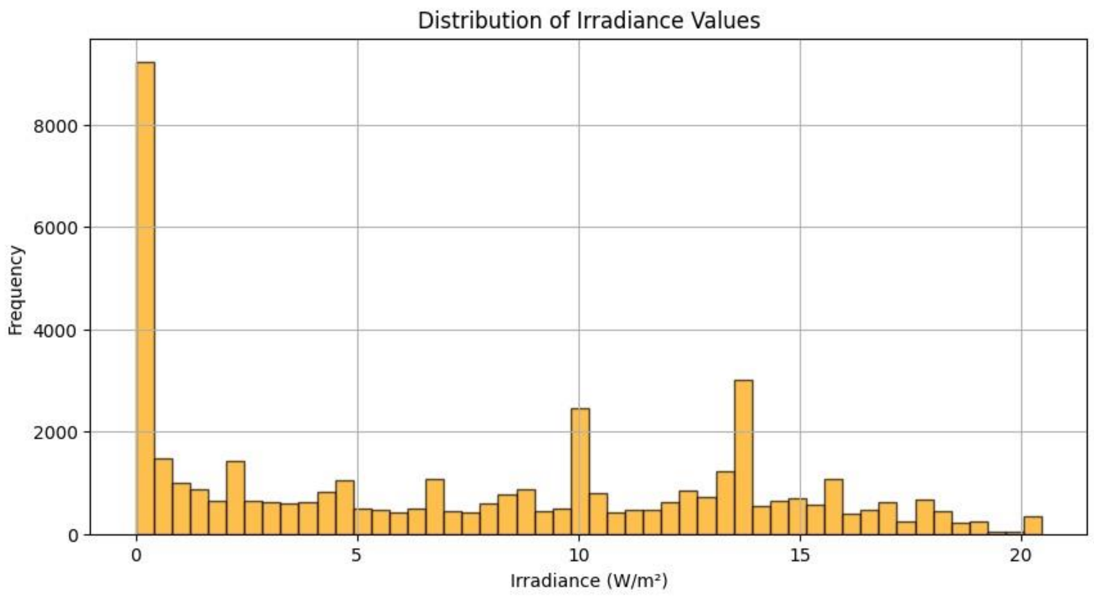
This scatter plot illustrates the relationship between solar irradiance (W/m²) and the total power output (kW) across all clusters. As expected, there is a positive nonlinear relationship power generation increases with irradiance initially but tends to plateau or slightly decline beyond a certain point. This may be due to environmental factors like temperature derating or system-level efficiency limits.



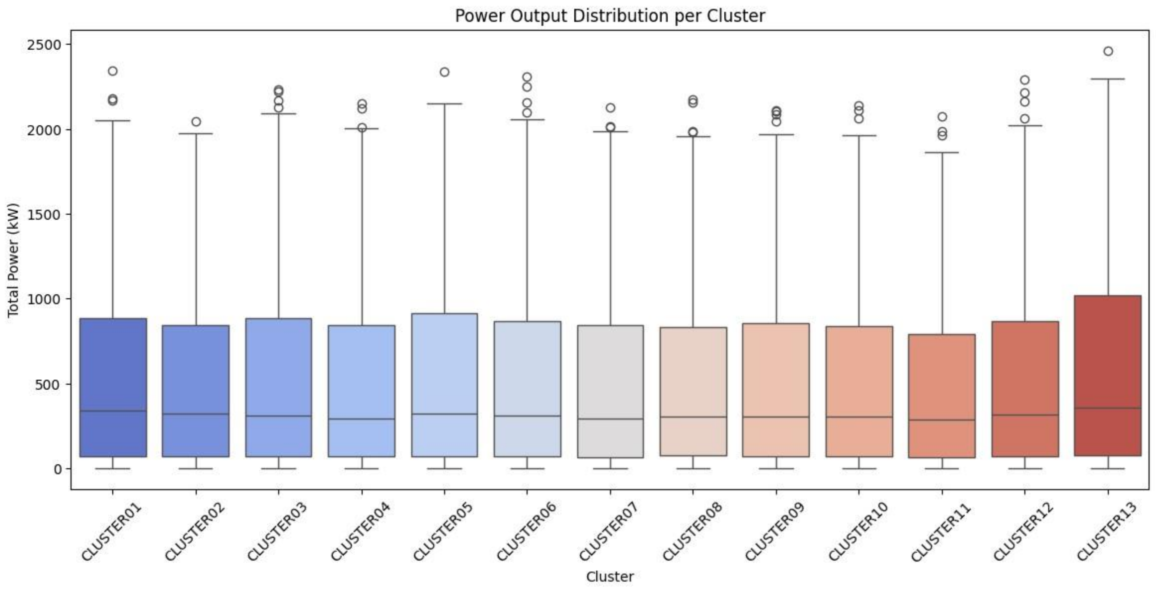
This horizontal bar chart shows the average daily total power output for each cluster. Clusters such as CLUSTER13 and CLUSTER05 exhibit higher average outputs, suggesting better performance or more favorable conditions. This comparison is useful for identifying underperforming clusters that may need maintenance or further investigation.



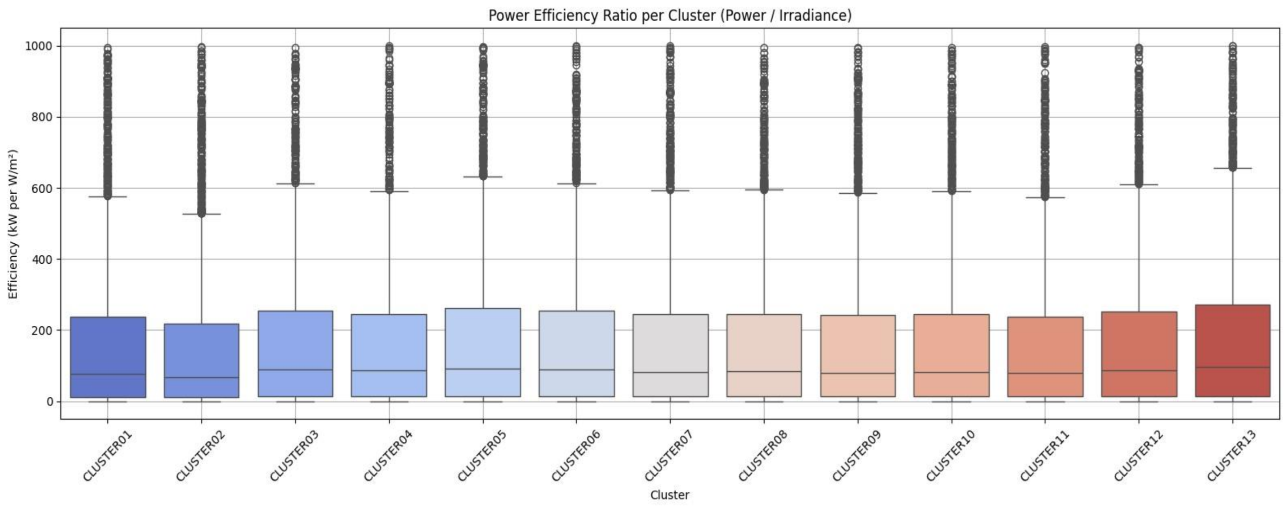
The line chart displays the total energy generated per day across all clusters. The fluctuations suggest variability in daily solar conditions, such as cloud cover, irradiance intensity, or equipment availability. Days with significantly lower output could indicate periods of poor weather or potential technical issues.



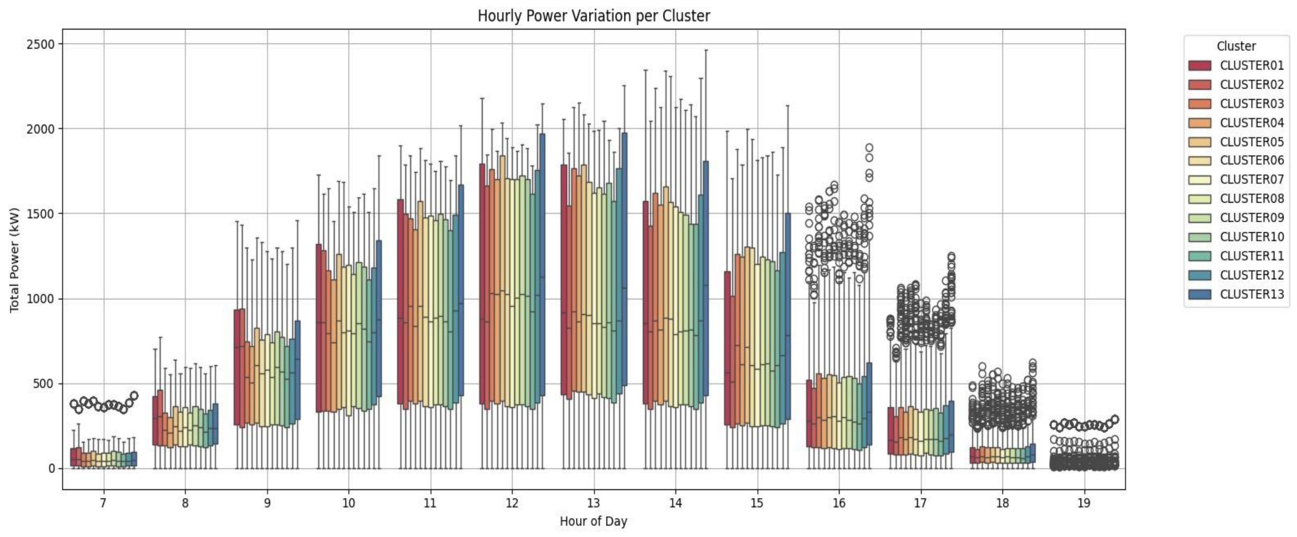
This histogram displays the frequency distribution of irradiance values (W/m²) collected throughout the month. A large portion of readings are clustered near zero, indicating either early morning, cloudy periods, or low-light conditions. There are noticeable peaks around typical daytime irradiance values, helping to understand the general weather patterns over time.



This boxplot presents the distribution of total power output (kW) for each cluster. It highlights the median, interquartile range, and outliers. While most clusters show similar variability, some (e.g., CLUSTER13) demonstrate higher median values and broader ranges, suggesting differences in equipment efficiency, orientation, or maintenance conditions.



This boxplot compares the power-to-irradiance efficiency across all clusters. It provides insights into how effectively each cluster converts sunlight into energy. Higher ratios indicate better efficiency. CLUSTER13, for example, appears to consistently outperform other clusters in terms of efficiency, while some clusters may have inefficiencies worth investigating.



This detailed boxplot shows how power output varies by hour across all clusters. The peak generation typically occurs between 11 AM and 2 PM, aligning with maximum sunlight intensity. Early morning and late evening hours show lower and more variable outputs. The plot also highlights performance consistency and helps identify potential anomalies or shading issues at specific hours.

2.3 Data Quality Assessment

The solar power dataset collected from the 30-day monitoring period was generally well-structured and organized. Each folder contained consistent files for inverter data, irradiance, substation output, and performance ratios. The data was mostly complete, with records logged at regular 5-minute intervals for each cluster.

Minor missing values were observed in the inverter and weather data, especially during periods with naturally low or zero solar output, such as early mornings or late evenings. These were expected and do not indicate sensor failure. No abnormal or unexpected missing patterns were found.

Additionally, no duplicate records or structural inconsistencies were detected in the datasets. Column names were standardized, and the formatting across all days was consistent. All numerical values were within realistic and expected ranges.

In summary, the data quality is satisfactory and ready for analysis. Only minimal cleaning was needed, such as handling null values and renaming a few columns for consistency. These small issues were resolved during preprocessing and did not impact the overall analysis.

2.4 Key Variables

Based on the initial exploration and the project objectives, several key variables have been identified as most relevant for understanding and predicting solar energy production and system performance:

Variable	**Description**	**Type**	**Role**
Irradiance	Solar energy received per unit area (W/m²), used to estimate power generation potential.	Numerical	Independent Variable
Total Power (kW)	Actual power output from inverters (kW), recorded every 5 minutes.	Numerical	Dependent Variable
Cluster	Grouping of solar panels/inverters (e.g., Cluster 1–13), used to analyze distributed performance.	Categorical	Group Identifier
Timestamp	Time of each observation (e.g., 13:15:00), enables temporal analysis.	DateTime	Time Reference
Hour	Extracted from timestamp to represent time of day for trend analysis.	Categorical	Feature Variable

Efficiency	Computed as Power / Irradiance, used to assess energy conversion performance.	Numerical	Derived Metric
Performance Ratio (PR)	Efficiency metric reflecting actual performance vs expected output, from PR.xlsx.	Numerical	Performance Metric

These variables form the basis for both descriptive analytics (EDA) and predictive modeling (e.g., Random Forest for power prediction, and XGBoost for performance classification).

2.5 Modeling Goals

The primary objective of the modeling phase is to develop accurate and insightful machine learning models that can support decision-making in solar energy performance management. Based on the initial data exploration and understanding of the domain, the project sets two key modeling goals:

I. **Power Output Prediction (Regression)**
To build a regression model capable of predicting the expected power output (in kilowatts) based on key influencing variables such as solar irradiance, time of day, and cluster identifier. This model helps estimate expected generation and detect potential underperformance.

II. **Cluster Performance Classification (Binary Classification)**
To develop a classification model that categorizes cluster performance as either High or Low, based on calculated efficiency metrics. The model enables stakeholders to monitor performance at the cluster level and proactively address inefficiencies.

Both models aim to enhance operational awareness, support timely interventions, and align with broader sustainability goals by improving solar energy utilization. Evaluation metrics such as R^2 Score, Mean Squared Error (MSE) for regression, and AUC, Confusion Matrix for classification are used to assess model performance

PHASE 3: DATA PREPARATION

3.1 File Structure and Sources

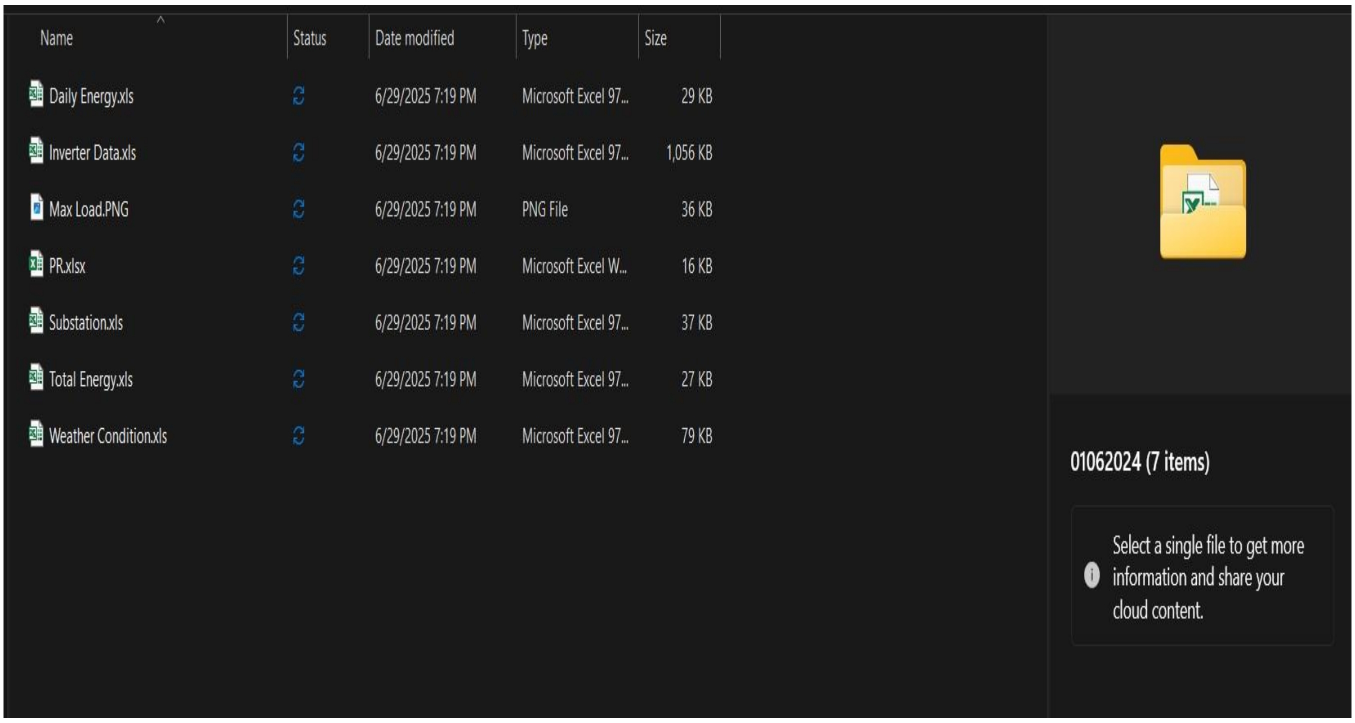
The primary dataset used in this project was obtained from the folder titled "2407_Weather Station (Jun24)", which contains data for 30 consecutive days in June 2024. Each day is represented as a subfolder named in the format DDMMYYYY (e.g., *01062024*, *02062024*, etc.). Inside every daily folder, there are several Excel files representing different aspects of solar power operations.

Among these files, two were particularly crucial for this project:

I. Inverter Data.xls, which contains 13 sheets (from *CLUSTER 1* to *CLUSTER 13*). Each sheet records active power generation from multiple inverters every 5 minutes.

II. Weather Condition.xls, which holds weather-related measurements, especially the Average Irradiance, reported at regular intervals.

Other files such as PR.xlsx, Substation.xls, and Total Energy.xls were available for reference but not directly used in the modeling pipeline.



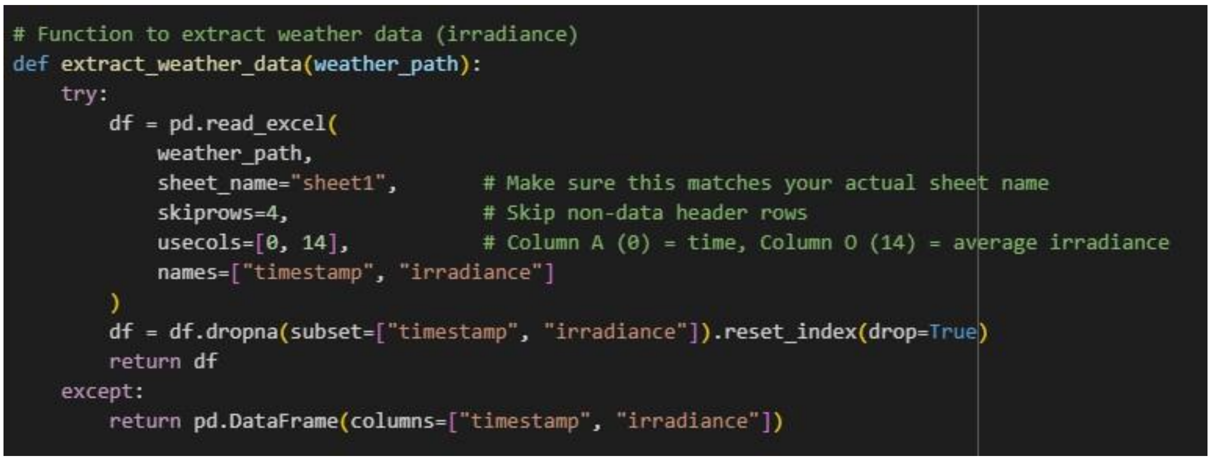
Name	Status	Date modified	Type	Size
Daily Energy.xls		6/29/2025 7:19 PM	Microsoft Excel 97...	29 KB
Inverter Data.xls		6/29/2025 7:19 PM	Microsoft Excel 97...	1,056 KB
Max Load.PNG		6/29/2025 7:19 PM	PNG File	36 KB
PR.xlsx		6/29/2025 7:19 PM	Microsoft Excel W...	16 KB
Substation.xls		6/29/2025 7:19 PM	Microsoft Excel 97...	37 KB
Total Energy.xls		6/29/2025 7:19 PM	Microsoft Excel 97...	27 KB
Weather Condition.xls		6/29/2025 7:19 PM	Microsoft Excel 97...	79 KB

01062024 (7 items)

Select a single file to get more information and share your cloud content.

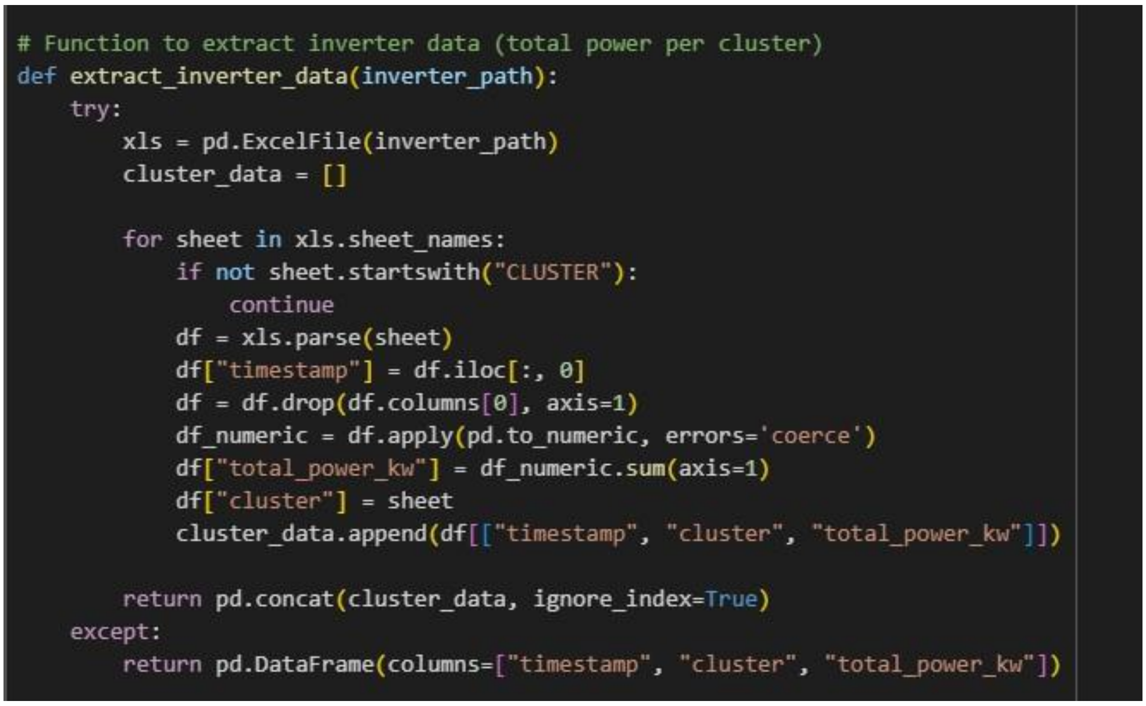
3.2 Data Merging and Consolidation

To transform the raw daily files into one unified dataset, a systematic merging approach was followed. Two custom functions were defined:



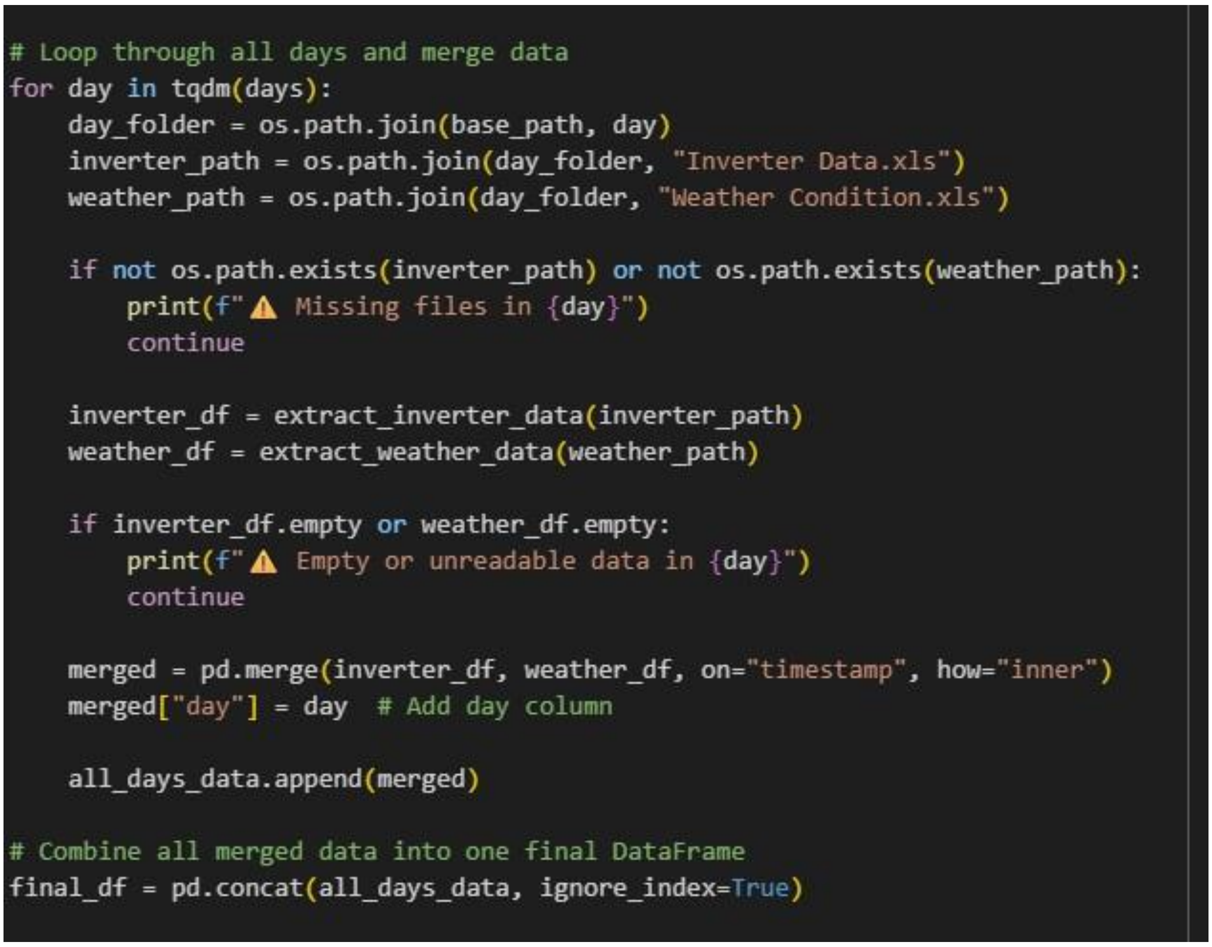
```python
# Function to extract weather data (irradiance)
def extract_weather_data(weather_path):
    try:
        df = pd.read_excel(
            weather_path,
            sheet_name="sheet1",       # Make sure this matches your actual sheet name
            skiprows=4,                # Skip non-data header rows
            usecols=[0, 14],           # Column A (0) = time, Column O (14) = average irradiance
            names=["timestamp", "irradiance"]
        )
        df = df.dropna(subset=["timestamp", "irradiance"]).reset_index(drop=True)
        return df
    except:
        return pd.DataFrame(columns=["timestamp", "irradiance"])
```

extract_weather_data () which reads irradiance values from column O in the Weather Condition.xls file while skipping header rows and cleaning missing values.



```python
# Function to extract inverter data (total power per cluster)
def extract_inverter_data(inverter_path):
    try:
        xls = pd.ExcelFile(inverter_path)
        cluster_data = []

        for sheet in xls.sheet_names:
            if not sheet.startswith("CLUSTER"):
                continue
            df = xls.parse(sheet)
            df["timestamp"] = df.iloc[:, 0]
            df = df.drop(df.columns[0], axis=1)
            df_numeric = df.apply(pd.to_numeric, errors='coerce')
            df["total_power_kw"] = df_numeric.sum(axis=1)
            df["cluster"] = sheet
            cluster_data.append(df[["timestamp", "cluster", "total_power_kw"]])

        return pd.concat(cluster_data, ignore_index=True)
    except:
        return pd.DataFrame(columns=["timestamp", "cluster", "total_power_kw"])
```

extract_inverter_data () which loops through all *CLUSTER* sheets in Inverter Data.xls, sums up the total active power per timestamp, labels each row with its cluster, and returns a combined DataFrame for the day.

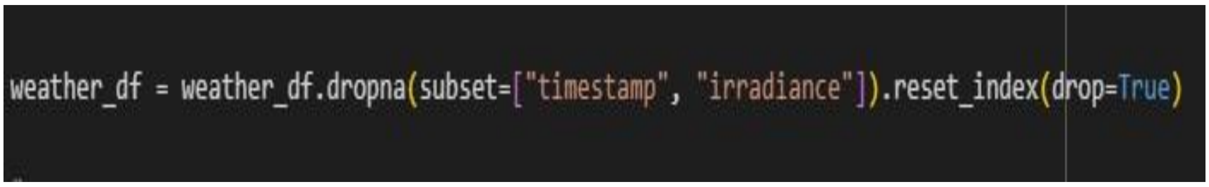


```python
# Loop through all days and merge data
for day in tqdm(days):
    day_folder = os.path.join(base_path, day)
    inverter_path = os.path.join(day_folder, "Inverter Data.xls")
    weather_path = os.path.join(day_folder, "Weather Condition.xls")

    if not os.path.exists(inverter_path) or not os.path.exists(weather_path):
        print(f"⚠ Missing files in {day}")
        continue

    inverter_df = extract_inverter_data(inverter_path)
    weather_df = extract_weather_data(weather_path)

    if inverter_df.empty or weather_df.empty:
        print(f"⚠ Empty or unreadable data in {day}")
        continue

    merged = pd.merge(inverter_df, weather_df, on="timestamp", how="inner")
    merged["day"] = day  # Add day column

    all_days_data.append(merged)

# Combine all merged data into one final DataFrame
final_df = pd.concat(all_days_data, ignore_index=True)
```

A loop was then used to iterate through all 30 folders. For each day, inverter and weather data were loaded, merged on the common column timestamp, and labeled with the corresponding day. Only valid merged datasets were appended to the final list. After processing all days, the results were concatenated into one large DataFrame and saved to a CSV file for future modeling.

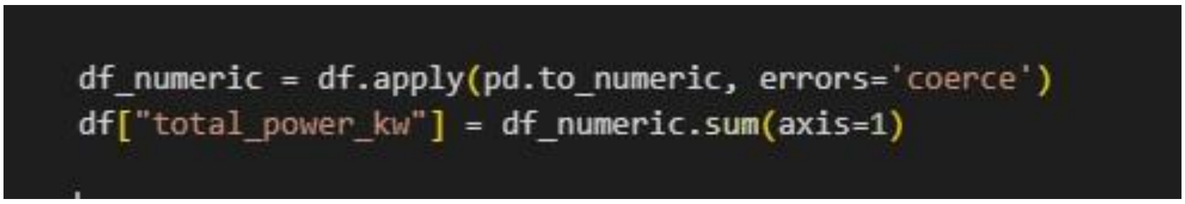
3.3 Data Cleaning

Several cleaning steps were applied to ensure data quality and consistency:



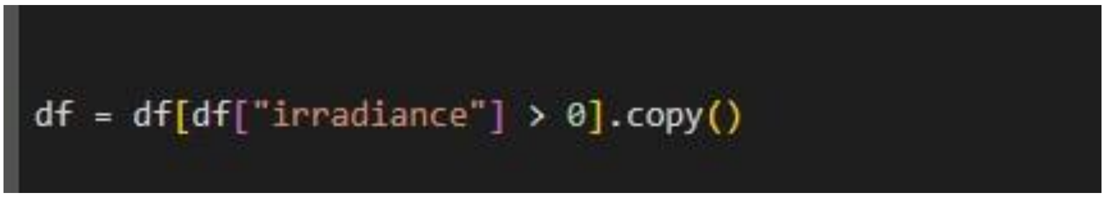
```
weather_df = weather_df.dropna(subset=["timestamp", "irradiance"]).reset_index(drop=True)
```

Rows containing missing values in critical fields such as timestamp or irradiance were dropped early using dropna.



```
df_numeric = df.apply(pd.to_numeric, errors='coerce')
df["total_power_kw"] = df_numeric.sum(axis=1)
```

Power readings were converted to numeric values with pd.to_numeric(errors='coerce'), allowing any non-numeric entries to be treated as NaNs.



```
df = df[df["irradiance"] > 0].copy()
```

To ensure that the data only includes valid records with actual solar irradiance, we filtered out all rows where the irradiance value is zero or negative. This step helps in removing periods when there was no sunlight (e.g., during the night or sensor errors), which would otherwise distort the analysis of solar efficiency.

3.4 Feature Engineering

After data consolidation and cleaning, new variables were created to enrich the dataset for analysis and modeling:

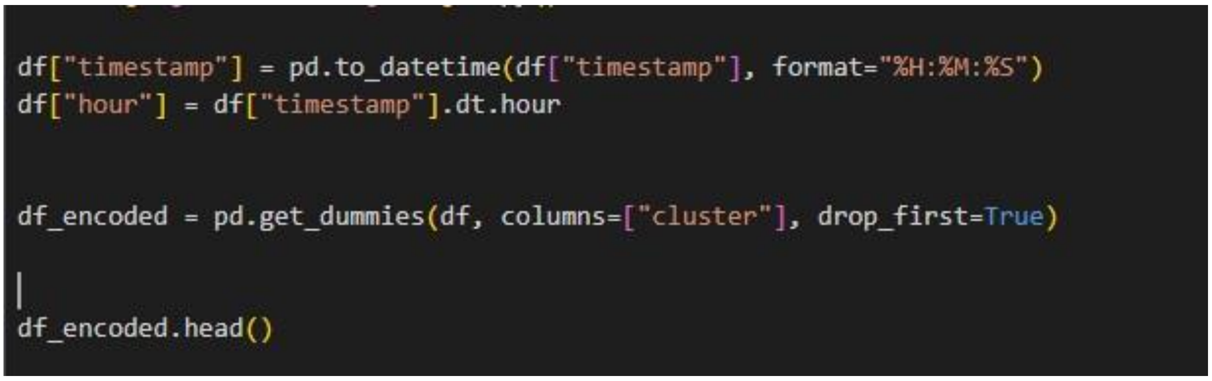
I. cluster: added based on the sheet name (e.g., CLUSTER 1, CLUSTER 2).

II. day: extracted from the folder name and added as a new column to distinguish daily records.

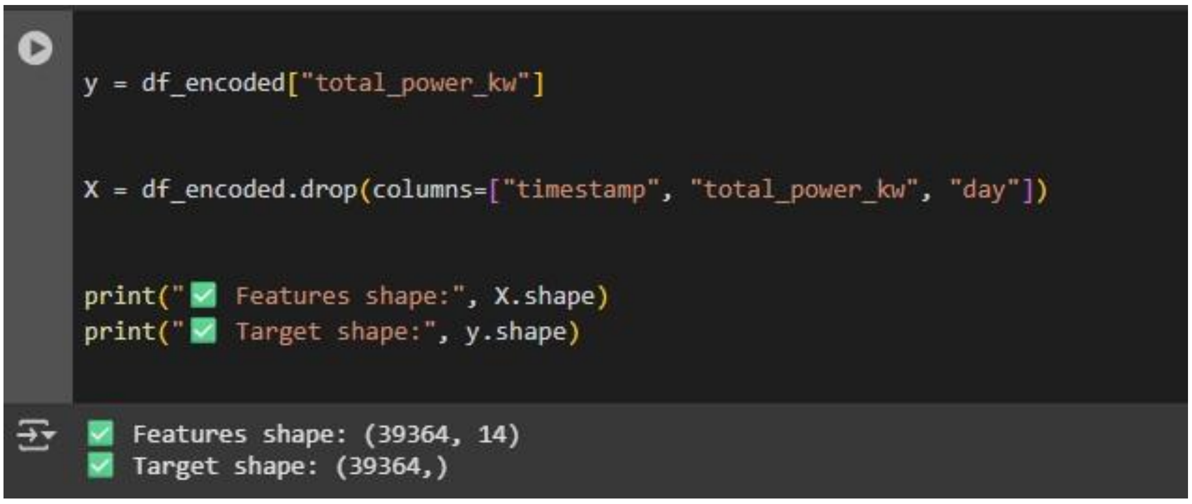
III. efficiency: computed later as the ratio between total_power_kw and irradiance to assess cluster performance.

IV. hour: extracted from the timestamp using .dt.hour to enable time-based grouping and visualization.

These engineered features were crucial in building hourly trends, efficiency comparisons, and classification models.



```
df["timestamp"] = pd.to_datetime(df["timestamp"], format="%H:%M:%S")
df["hour"] = df["timestamp"].dt.hour


df_encoded = pd.get_dummies(df, columns=["cluster"], drop_first=True)


df_encoded.head()
```



```
y = df_encoded["total_power_kw"]


X = df_encoded.drop(columns=["timestamp", "total_power_kw", "day"])


print("✅ Features shape:", X.shape)
print("✅ Target shape:", y.shape)
```

```
✅ Features shape: (39364, 14)
✅ Target shape: (39364,)
```

3.5 Final Dataset & Export

The final dataset is a consolidated and cleaned table containing 44,330 rows of synchronized 5-minute readings across multiple clusters. Each row represents a specific timepoint and includes the following key variables: I. timestamp: Time of measurement.

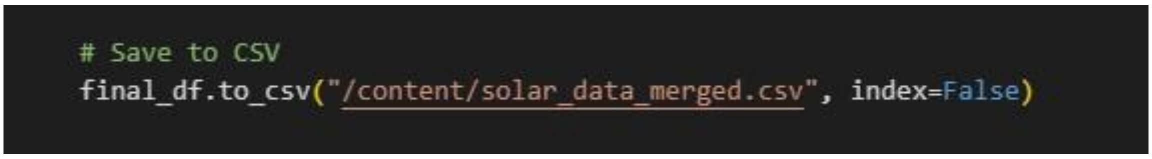
II. cluster: The solar cluster (e.g., CLUSTER01) from which the reading was recorded.

III. total_power_kw: The total AC power output from that cluster at the specific timestamp.

IV. irradiance: The average solar irradiance at the same timestamp, obtained from the weather station.

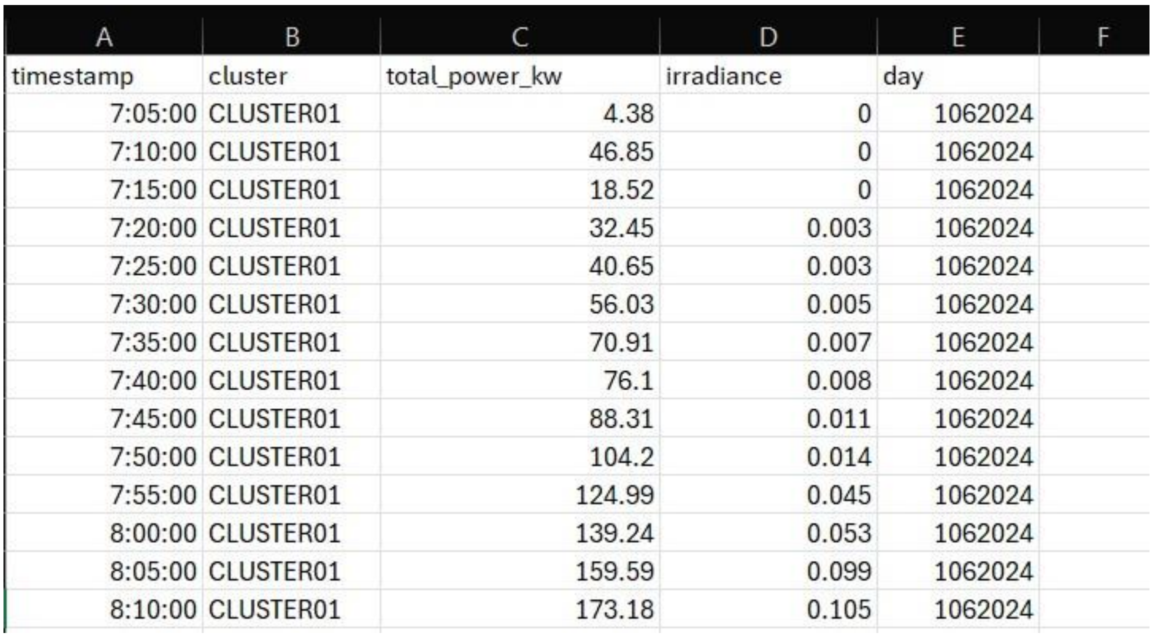
V. day: A numeric identifier for the date (e.g., 1062024 represents June 1, 2024).

This dataset merges both inverter data and weather readings for 30 consecutive days. It serves as the essential foundation for all subsequent analysis, including modeling and visual dashboards.



```
# Save to CSV
final_df.to_csv("/content/solar_data_merged.csv", index=False)
```

The cleaned dataset was saved as solar_data_merged.csv, ready for the modeling phase.



A	B	C	D	E	F
timestamp	cluster	total_power_kw	irradiance	day	
7:05:00	CLUSTER01	4.38	0	1062024	
7:10:00	CLUSTER01	46.85	0	1062024	
7:15:00	CLUSTER01	18.52	0	1062024	
7:20:00	CLUSTER01	32.45	0.003	1062024	
7:25:00	CLUSTER01	40.65	0.003	1062024	
7:30:00	CLUSTER01	56.03	0.005	1062024	
7:35:00	CLUSTER01	70.91	0.007	1062024	
7:40:00	CLUSTER01	76.1	0.008	1062024	
7:45:00	CLUSTER01	88.31	0.011	1062024	
7:50:00	CLUSTER01	104.2	0.014	1062024	
7:55:00	CLUSTER01	124.99	0.045	1062024	
8:00:00	CLUSTER01	139.24	0.053	1062024	
8:05:00	CLUSTER01	159.59	0.099	1062024	
8:10:00	CLUSTER01	173.18	0.105	1062024	

PHASE 4: MODELING

4.1 Overview of Modelling Strategy

In this project, the modelling phase was designed to extract predictive insights from the processed solar dataset. Two main machine learning tasks were carried out, each serving a different purpose within the solar power monitoring system. The first task focused on predicting the actual power output of each cluster based on environmental and temporal features, while the second aimed to classify the performance level of each cluster (High or Low) using an efficiency metric derived from the data.

These models were developed using Python and trained on the merged dataset generated during the data preparation phase. The predictions produced by both models enable proactive monitoring and forecasting, which are crucial for improving energy yield and identifying underperforming units in real time.

4.2 Model 1: Power Output Prediction

The first model focused on estimating the expected power output (in kilowatts) of each solar cluster based on environmental and temporal inputs. This is framed as a regression task, where the aim is to predict a continuous value using features such as irradiance, cluster identity, and time of day.

Algorithms Used:

To achieve this, three regression algorithms were explored and compared: Logistic Regression, Random Forest, and XGBoost.

Logistic Regression was initially used as a simple baseline, even though it is more suited for classification. As expected, its performance was limited due to the complex, nonlinear nature of the data.

Next, Random Forest Regressor was implemented. Random Forest is an ensemble of decision trees that combines multiple models to improve overall prediction accuracy. It handles non-linear relationships well and reduces the risk of overfitting. Among all tested models, Random Forest produced the best predictive performance, offering both stability and accuracy across the dataset.

XGBoost Regressor was also tested. While it is a high-performance gradient boosting algorithm often known for excellent results, in this particular case, its predictions were slightly less accurate than those from Random Forest. Nonetheless, it remains a strong alternative

Features Used:

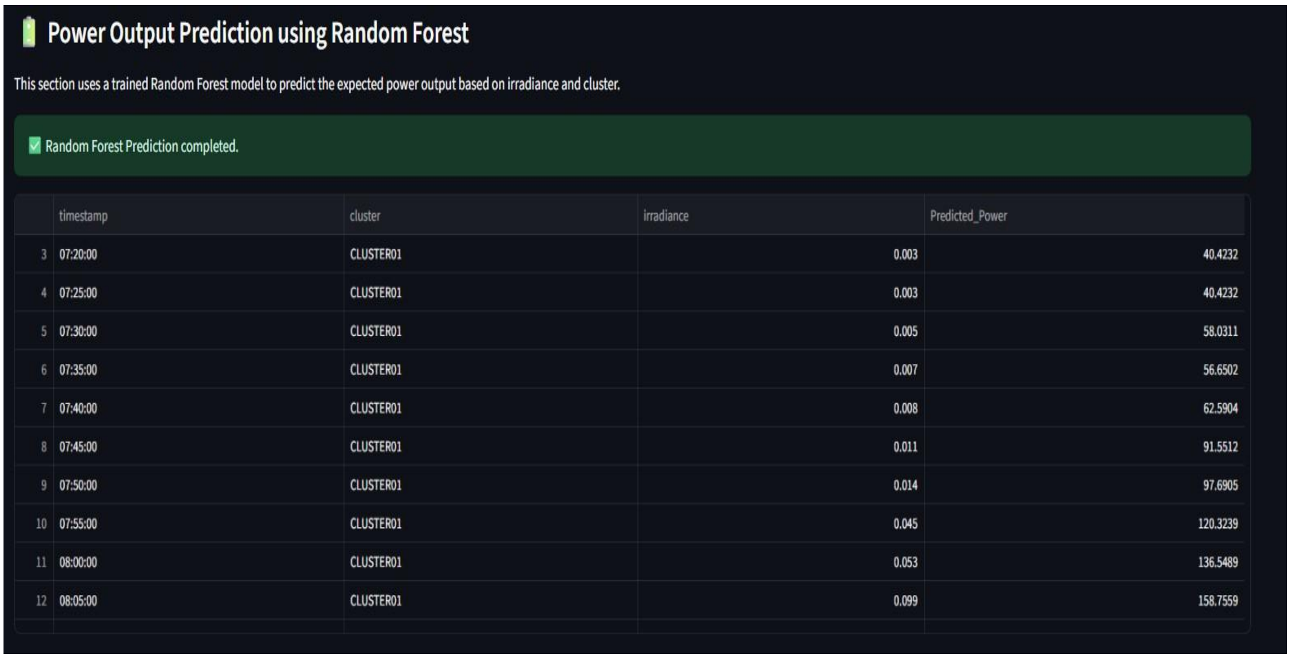
I. irradiance – measures the solar energy available at that moment.

II. cluster – indicates which solar cluster the data belongs to.

III. hour – extracted from the timestamp to capture time-based variation in solar production

Target Variable and Output:

while the target was total_power_kw. The output from this model provides a continuous prediction of expected power generation for each time interval and cluster.



Power Output Prediction using Random Forest

This section uses a trained Random Forest model to predict the expected power output based on irradiance and cluster.

✅ Random Forest Prediction completed.

	timestamp	cluster	irradiance	Predicted_Power
3	07:20:00	CLUSTER01	0.003	40.4232
4	07:25:00	CLUSTER01	0.003	40.4232
5	07:30:00	CLUSTER01	0.005	58.0311
6	07:35:00	CLUSTER01	0.007	56.6502
7	07:40:00	CLUSTER01	0.008	62.5904
8	07:45:00	CLUSTER01	0.011	91.5512
9	07:50:00	CLUSTER01	0.014	97.6905
10	07:55:00	CLUSTER01	0.045	120.3239
11	08:00:00	CLUSTER01	0.053	136.5489
12	08:05:00	CLUSTER01	0.099	158.7559

4.3 Model 2: Cluster Performance Classification

The second model aimed to classify the operational performance of each solar cluster as either *High* or *Low*. This framing turns the task into a binary classification problem, where each data point is labeled based on how efficient the cluster was at a given time, considering the solar irradiance levels. **Algorithms Used:**

To perform this classification, the powerful XGBoost Classifier was used. XGBoost is an optimized implementation of gradient boosting designed for high performance and scalability. It is particularly effective for structured data and was chosen due to its ability to handle non-linear interactions and provide high accuracy with relatively low training time.

The model learns patterns that distinguish high-performing clusters from lowperforming ones by analyzing variations in efficiency throughout the day, across different clusters, and under changing irradiance conditions.

Features Used:

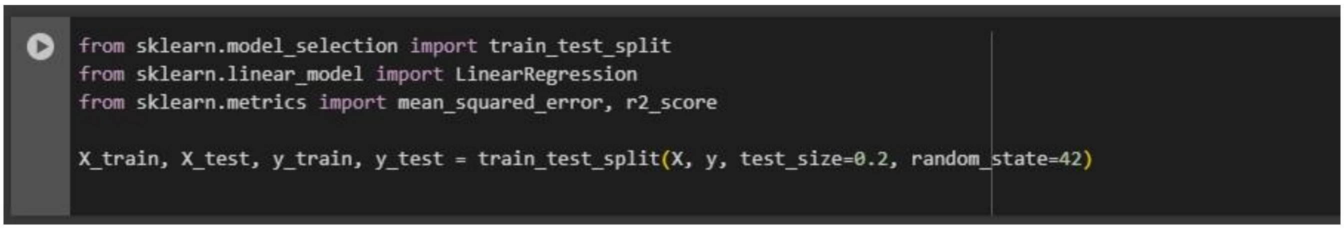
1.irradiance – the level of solar energy available, crucial for understanding expected output.
2. hour – extracted from the timestamp to capture time-dependent patterns in performance.
3.cluster (one-hot encoded) – represents the cluster identity, allowing the model to learn cluster-specific behaviors and deviations.

Target Variable and Output:

The target variable was a binary label named performance_label, where 1 indicates a *High* performing cluster and 0 indicates *Low* performance. These labels were derived by comparing each cluster's calculated efficiency (i.e., total_power_kw / irradiance) against the median efficiency across the dataset.

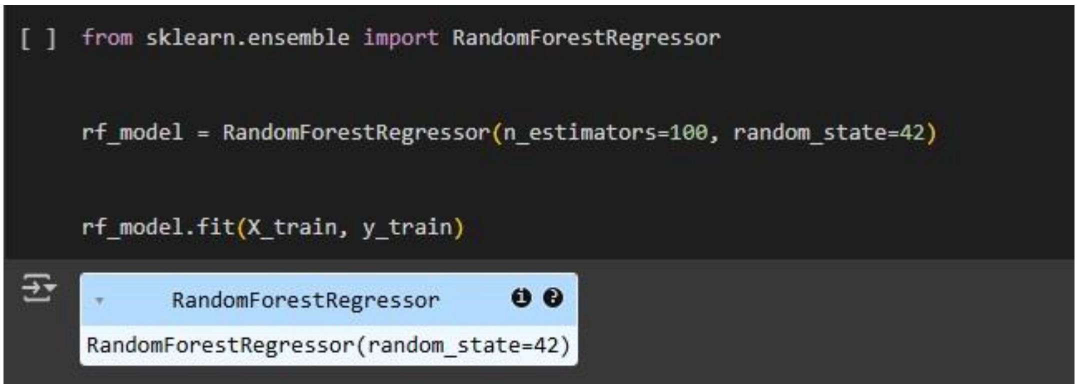
The output of this model enables stakeholders to quickly detect underperforming clusters in real-time, providing valuable insights for maintenance planning and operational optimization.

4.4 Data Splitting and Model Training



```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

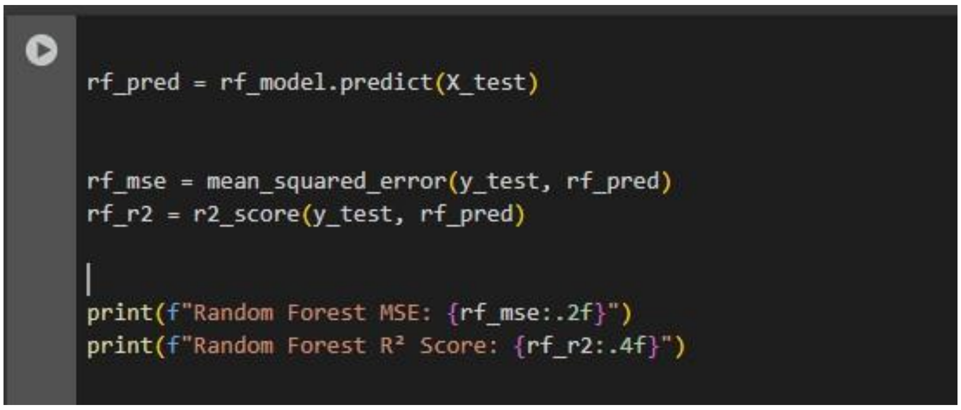
This image shows the use of the train_test_split function from sklearn.model_selection to divide the dataset into training (80%) and testing (20%) sets. The random_state=42 ensures reproducibility of the split.



```python
[ ]  from sklearn.ensemble import RandomForestRegressor


     rf_model = RandomForestRegressor(n_estimators=100, random_state=42)


     rf_model.fit(X_train, y_train)
```

```
RandomForestRegressor
RandomForestRegressor(random_state=42)
```

This snippet initializes a RandomForestRegressor with 100 estimators and a fixed random seed. The model is then trained using the fit() function on the training data (X_train, y_train).



```python
rf_pred = rf_model.predict(X_test)


rf_mse = mean_squared_error(y_test, rf_pred)
rf_r2 = r2_score(y_test, rf_pred)


print(f"Random Forest MSE: {rf_mse:.2f}")
print(f"Random Forest R² Score: {rf_r2:.4f}")
```

This part of the code performs prediction using the trained model on the test set (X_test). It then evaluates the model using Mean Squared Error (MSE) and R² Score, followed by printing the results for review.

PHASE 5: EVALUATION AND RESULTS

5.1 Introduction

This chapter presents the evaluation of the machine learning models used in the solar power analysis project. Each model was assessed using appropriate metrics and visualizations to determine its performance and reliability. The chapter is divided into two main sections: the regression model for predicting power output, and the classification model for evaluating cluster performance.

5.2 Power Output Prediction Evaluation

The first set of models focused on predicting the total power output (kW) based on features such as irradiance, cluster, and hour. The goal was to estimate a continuous numeric value, making this a regression task.

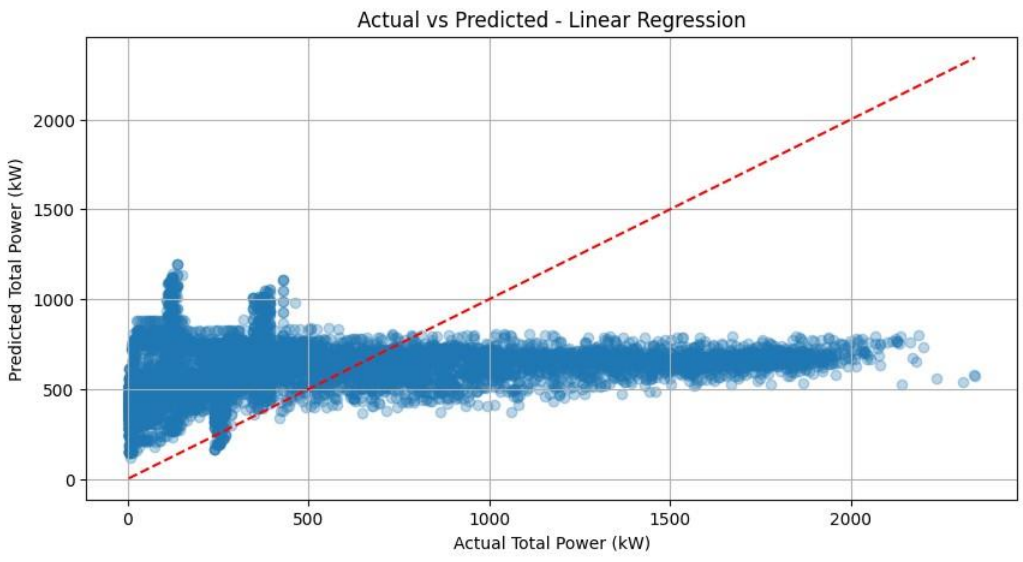
Evaluation Metrics Used:

To evaluate model accuracy, the following metrics were used:

I. Mean Squared Error (MSE): Measures the average squared difference between actual and predicted values.
II. R^2 Score (Coefficient of Determination): Indicates how well the predictions match the true values (1.0 = perfect match).

Model Results:

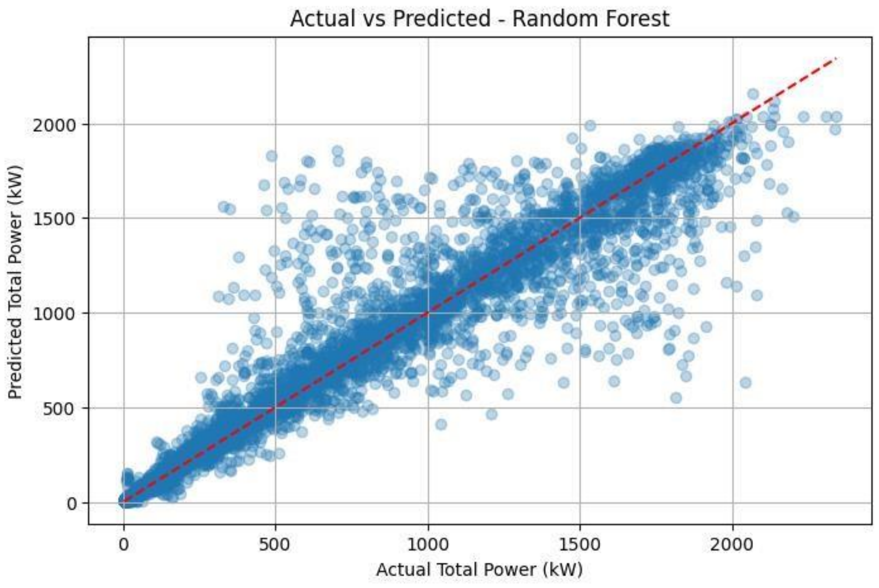
- **Linear Regression**
  - MSE: 288,760.64
  - R^2 Score: 0.0666



As shown in the figure above predictions were poorly aligned with actual values, indicating the model's limited ability to capture complex relationships.

Random Forest Regressor

- MSE: 24,080.23
- R² Score: 0.9222

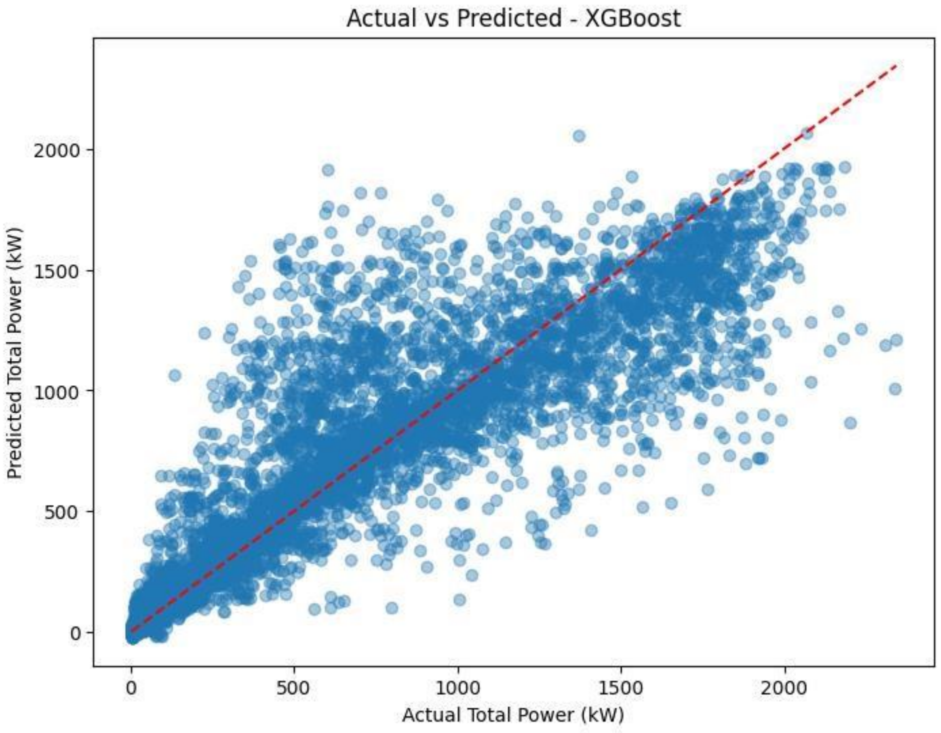


This model showed excellent performance, with predictions closely following the actual values, as seen in the figure above.

• XGBoost Regressor

- MSE: 57,336.22

- R² Score: 0.8147



Performed well, but slightly below Random Forest. Still a valid alternative with good predictive power.

Summary of Comparison:

Model	Type	MSE	R² Score	Performance Summary
Linear Regression	Regression	288,760.64	0.0666	Poor performance; unable to model non-linear patterns. Used only as a baseline.
Random Forest	Regression	24,080.23	0.9222	Best performing model. Captured non-linearities well. Stable and accurate.
XGBoost	Regression	57,336.22	0.8147	Strong model with good performance. Slightly less accurate than Random Forest.

5.3 Cluster Performance Classification Evaluation

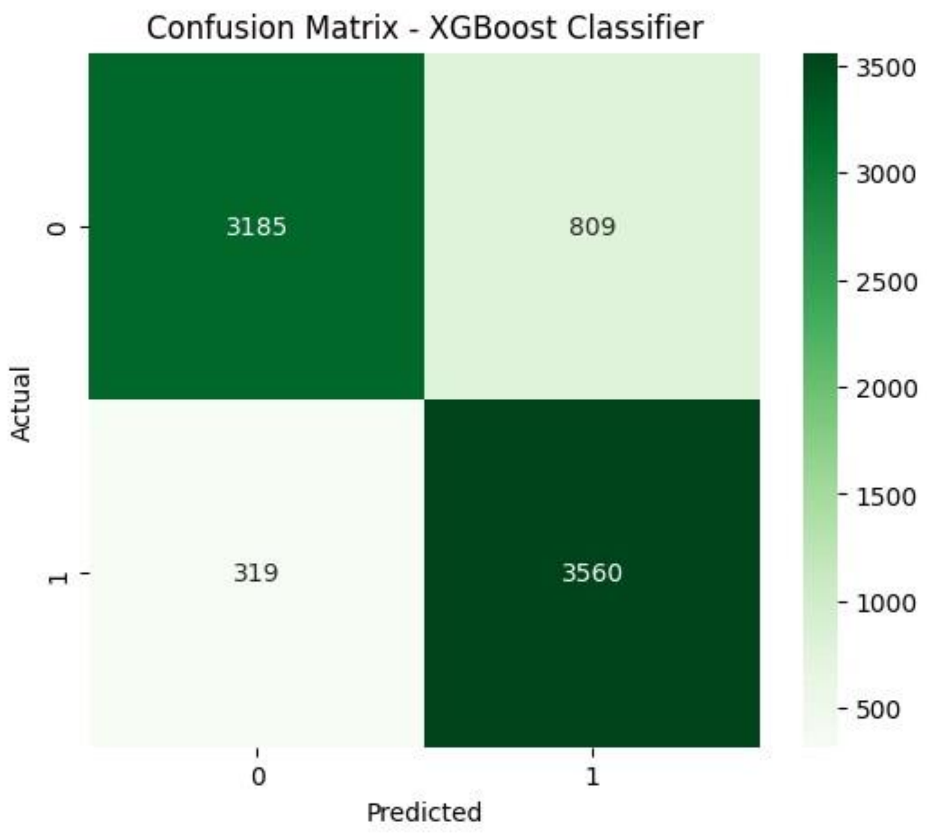
The second model aimed to classify whether a cluster's performance was High or Low based on its calculated efficiency. This is a binary classification task.

Evaluation Metrics Used

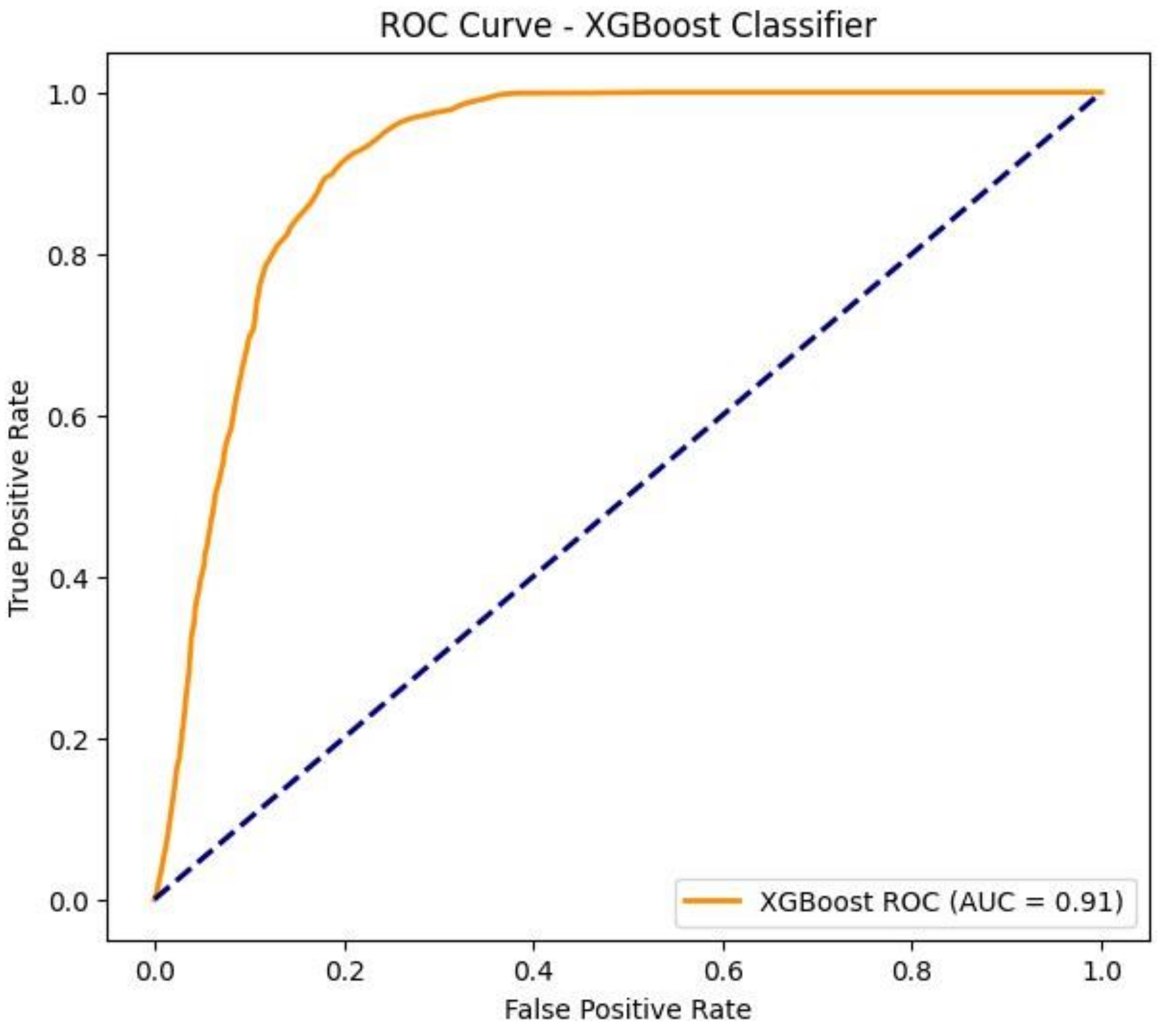
I. Confusion Matrix: Shows the number of correct and incorrect predictions for each class.
II. AUC-ROC Curve: Visualizes the trade-off between true positive rate and false positive rate, with the AUC indicating overall classification quality.

XGBoost Classifier Results

- Confusion Matrix:
  - True Positives: 3,560 ○ True Negatives: 3,185 ○ False Positives: 809 ○ False Negatives: 319



- ROC Curve and AUC Score:
  - AUC Score: 0.91 ○ The model demonstrated excellent ability to distinguish between High and Low performing clusters.



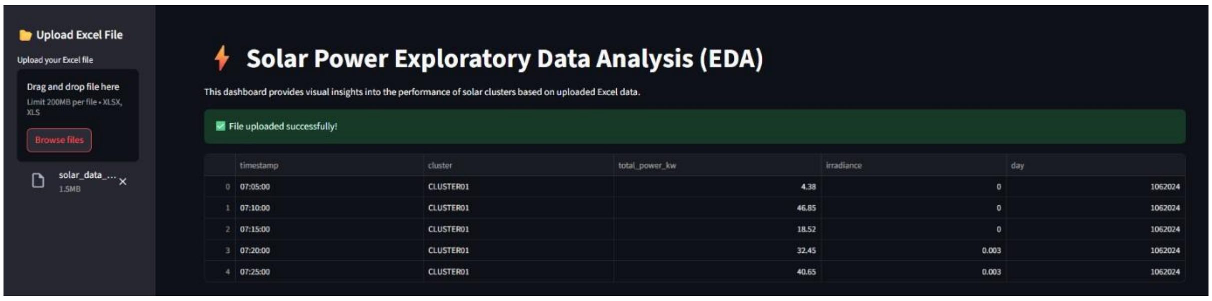
PHASE 6: DEPLOYMENT

6.1 Introduction

To demonstrate the practical value of the developed models and enable stakeholder interaction, a Streamlit-based web prototype was created. This interactive dashboard allows users to upload solar power data in Excel format and instantly explore both visual analytics and machine learning predictions.

6.2 Data Upload and Preview

Users can upload Excel files containing solar cluster data. Upon upload, the system performs initial validation and displays a preview of the first few rows for quick inspection**.**



Solar Power Exploratory Data Analysis (EDA)

This dashboard provides visual insights into the performance of solar clusters based on uploaded Excel data.

✓ File uploaded successfully!

	timestamp	cluster	total_power_kw	irradiance	day
0	07:05:00	CLUSTER01	4.38	0	1062024
1	07:10:00	CLUSTER01	46.85	0	1062024
2	07:15:00	CLUSTER01	18.52	0	1062024
3	07:20:00	CLUSTER01	32.45	0.003	1062024
4	07:25:00	CLUSTER01	40.65	0.003	1062024

6.3 Interactive EDA Visualizations

The dashboard offers several interactive plots to explore relationships in the data:

I. Scatter Plot showing how irradiance affects power output across all clusters.

II. Bar Chart displaying average total power per cluster.

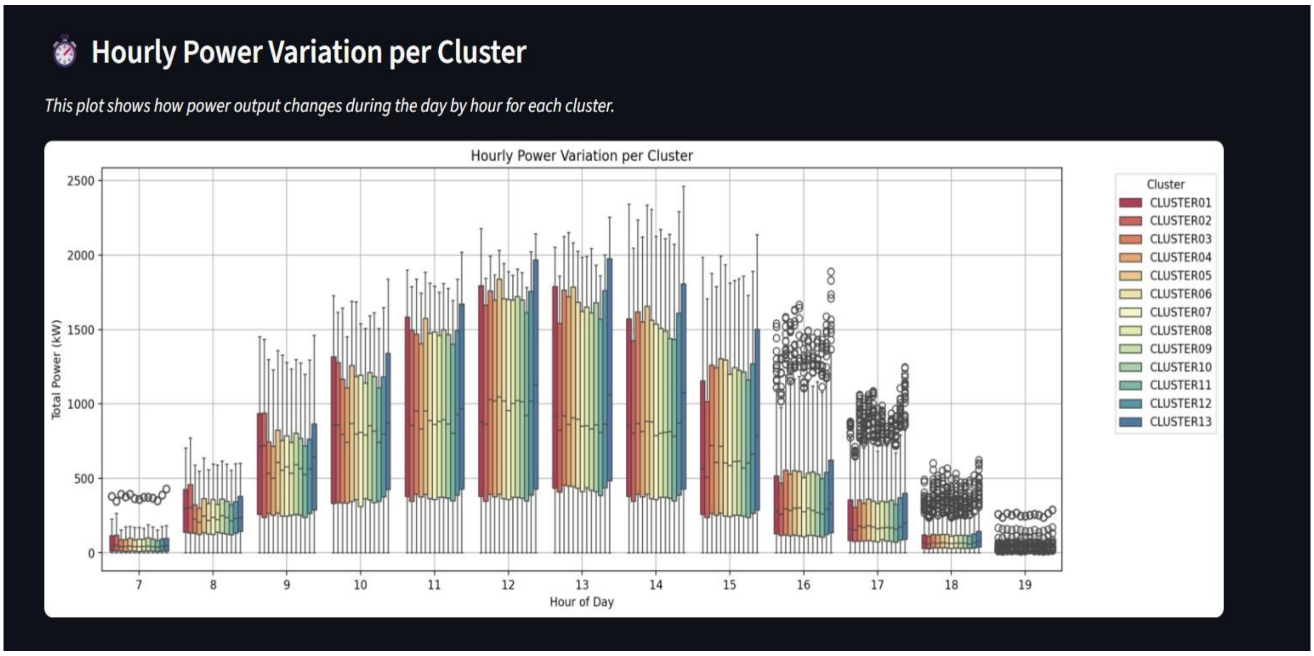
III. Line Chart representing total energy production across days.

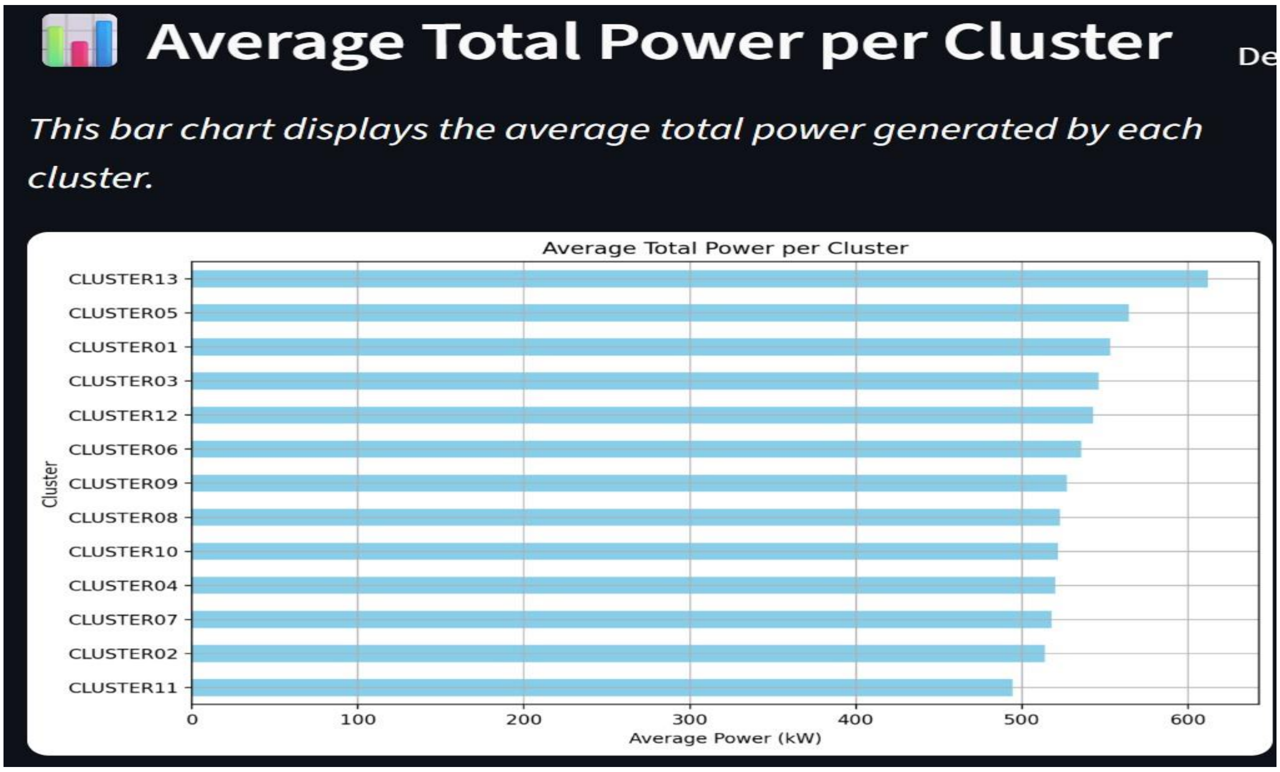
IV. Histogram showing the distribution of irradiance values.

V. Boxplot comparing power output across clusters.

VI. Efficiency Analysis to visualize how well clusters convert irradiance into power.

VII. Hourly Analysis of power variations throughout the day per cluster.



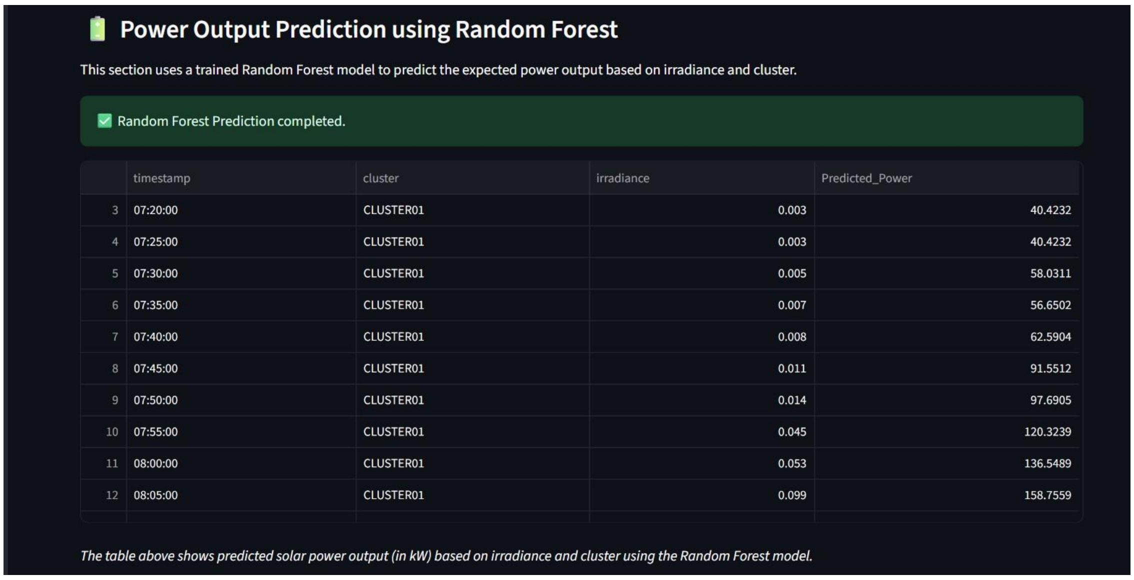


6.4 Models Output Prediction

The dashboard integrates two trained machine learning models for intelligent analysis of solar cluster performance:

- A Random Forest Regressor is used to predict the expected solar power output (in kilowatts) based on features like irradiance, cluster identity, and time. Users can view predicted power values and compare them with actual measurements to assess forecasting accuracy.

- An XGBoost Classifier is used to classify the operational efficiency of each cluster as either High or Low, based on the calculated efficiency ratio (power/irradiance). This allows users to monitor performance status across different clusters in real time.



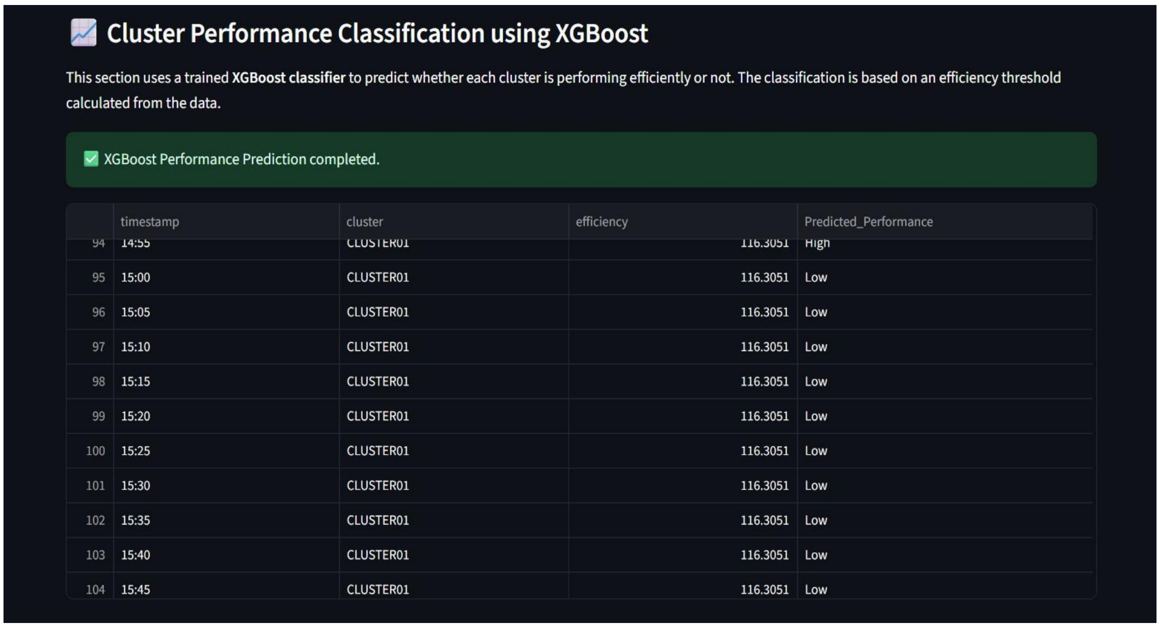
Power Output Prediction using Random Forest

This section uses a trained Random Forest model to predict the expected power output based on irradiance and cluster.

✓ Random Forest Prediction completed.

	timestamp	cluster	irradiance	Predicted_Power
3	07:20:00	CLUSTER01	0.003	40.4232
4	07:25:00	CLUSTER01	0.003	40.4232
5	07:30:00	CLUSTER01	0.005	58.0311
6	07:35:00	CLUSTER01	0.007	56.6502
7	07:40:00	CLUSTER01	0.008	62.5904
8	07:45:00	CLUSTER01	0.011	91.5512
9	07:50:00	CLUSTER01	0.014	97.6905
10	07:55:00	CLUSTER01	0.045	120.3239
11	08:00:00	CLUSTER01	0.053	136.5489
12	08:05:00	CLUSTER01	0.099	158.7559

The table above shows predicted solar power output (in kW) based on irradiance and cluster using the Random Forest model.



Cluster Performance Classification using XGBoost

This section uses a trained **XGBoost classifier** to predict whether each cluster is performing efficiently or not. The classification is based on an efficiency threshold calculated from the data.

✓ XGBoost Performance Prediction completed.

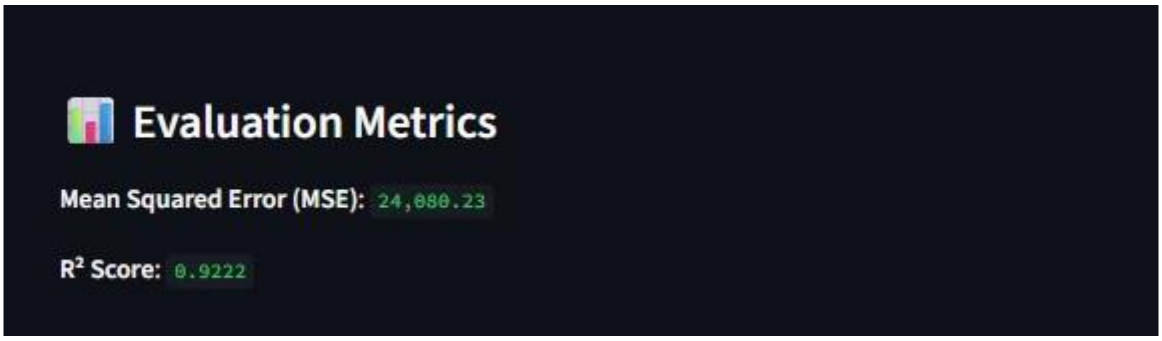
	timestamp	cluster	efficiency	Predicted_Performance
94	14:55	CLUSTER01	116.3051	High
95	15:00	CLUSTER01	116.3051	Low
96	15:05	CLUSTER01	116.3051	Low
97	15:10	CLUSTER01	116.3051	Low
98	15:15	CLUSTER01	116.3051	Low
99	15:20	CLUSTER01	116.3051	Low
100	15:25	CLUSTER01	116.3051	Low
101	15:30	CLUSTER01	116.3051	Low
102	15:35	CLUSTER01	116.3051	Low
103	15:40	CLUSTER01	116.3051	Low
104	15:45	CLUSTER01	116.3051	Low

6.5 Models Results

The dashboard also presents evaluation results for both machine learning models to help users assess their performance:

- For the Random Forest Regressor, a scatter plot compares actual vs predicted power output, along with key evaluation metrics such as Mean Squared Error (MSE) and R² Score to indicate regression accuracy.
- For the XGBoost Classifier, the dashboard displays a confusion matrix and ROC curve to evaluate classification accuracy, sensitivity, and overall model reliability in identifying high- and low-performing clusters.

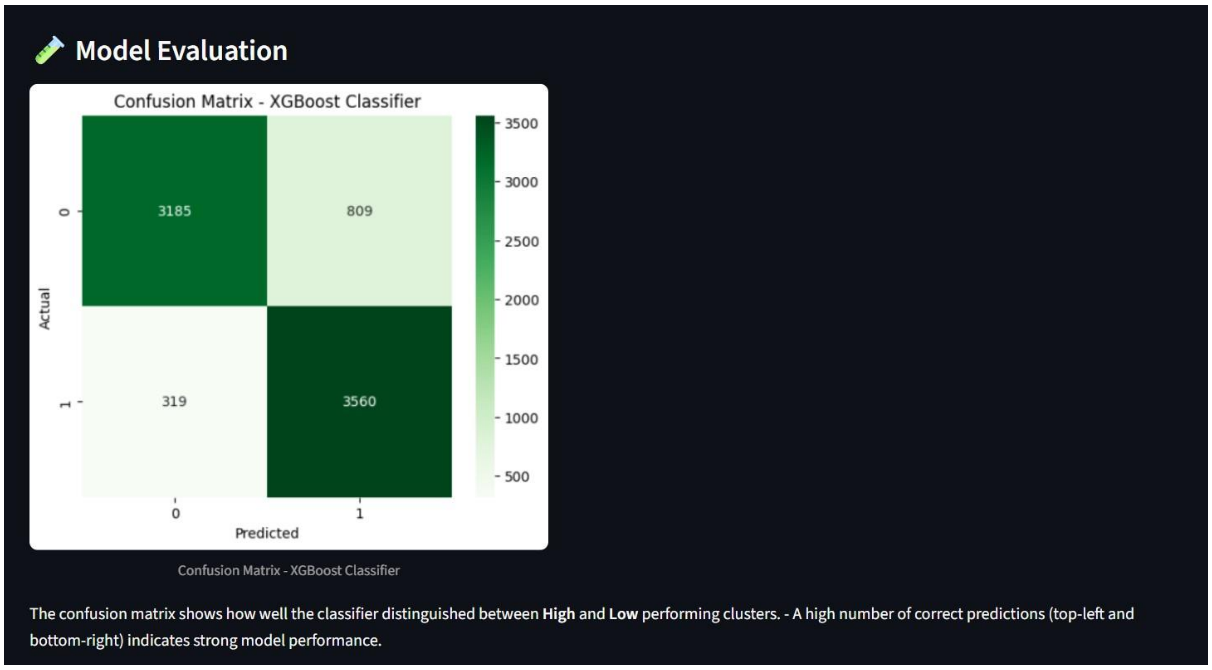
These evaluation tools provide valuable feedback for validating model effectiveness and building trust in the system's predictions.



Evaluation Metrics

Mean Squared Error (MSE): `24,080.23`

R² Score: `0.9222`



Model Evaluation

Confusion Matrix - XGBoost Classifier

The confusion matrix shows how well the classifier distinguished between **High** and **Low** performing clusters. - A high number of correct predictions (top-left and bottom-right) indicates strong model performance.

7. Conclusion and Future Recommendations

7.1 Conclusion

This project successfully demonstrated the potential of machine learning models in analyzing and predicting solar energy performance using real-world inverter and weather data. Two key models were developed and deployed within an interactive dashboard:

- A **Random Forest regression model** for predicting solar power output based on irradiance, cluster, and time features.
- An **XGBoost classification model** to assess cluster performance by identifying high vs. low efficiency periods.

The interactive dashboard allowed users to explore the data visually, view predictive insights, and understand evaluation results through meaningful charts and tables. These features provide valuable support for solar energy monitoring and operational planning.

7.2 Future Recommendations

To further enhance the solution and prepare it for full-scale deployment, the following improvements are recommended:

1. **Real-Time Data Integration**
   Connect the dashboard to live data streams from solar inverters and weather stations to enable real-time prediction and monitoring.

2. **Alert and Notification System**
   Incorporate automated alerts that notify operators when a cluster's performance drops below a certain threshold.

3. **Model Retraining with Seasonal Data**
   Continuously retrain the models with new data that includes seasonal and weather variations to maintain high accuracy across different conditions.

4. **Add More Predictive Features**
   Include additional features such as panel temperature, humidity, and wind speed, which could improve model accuracy further.

5. **Mobile-Friendly Deployment**
   Adapt the dashboard interface for mobile or tablet use, allowing operators to access the system remotely.

6. **Anomaly Detection Module**
   Develop a separate model to detect faults or anomalies in cluster behavior, such as sudden drops in efficiency or irregular patterns.

7. **Energy Forecast Reporting**
   Allow users to export scheduled forecasts or reports in PDF or Excel format for planning and documentation.

By implementing these improvements, the system can transition from a prototype to a robust, production-level solution capable of supporting large-scale solar infrastructure management.